



Usando FIWARE FIWARE ZONE_

<http://www.fiware.org>

<http://lab.fiware.org>

Follow @FIWARE on Twitter!



0. Conocimientos previos

1. C++
2. HTML
3. API REST
4. FIWARE – IoT Agent Ultralighth 2.0

01 >

Hardware

Specs

ESP8266

RAM: 128KB

ROM: 4MB

Alimentación: 3.3V

WiFi integrado

13 Pin IO

USB<->UART incluido

Regulador 5V (USB) a 3.3V



ESP-12E DEVELOPMENT BOARD

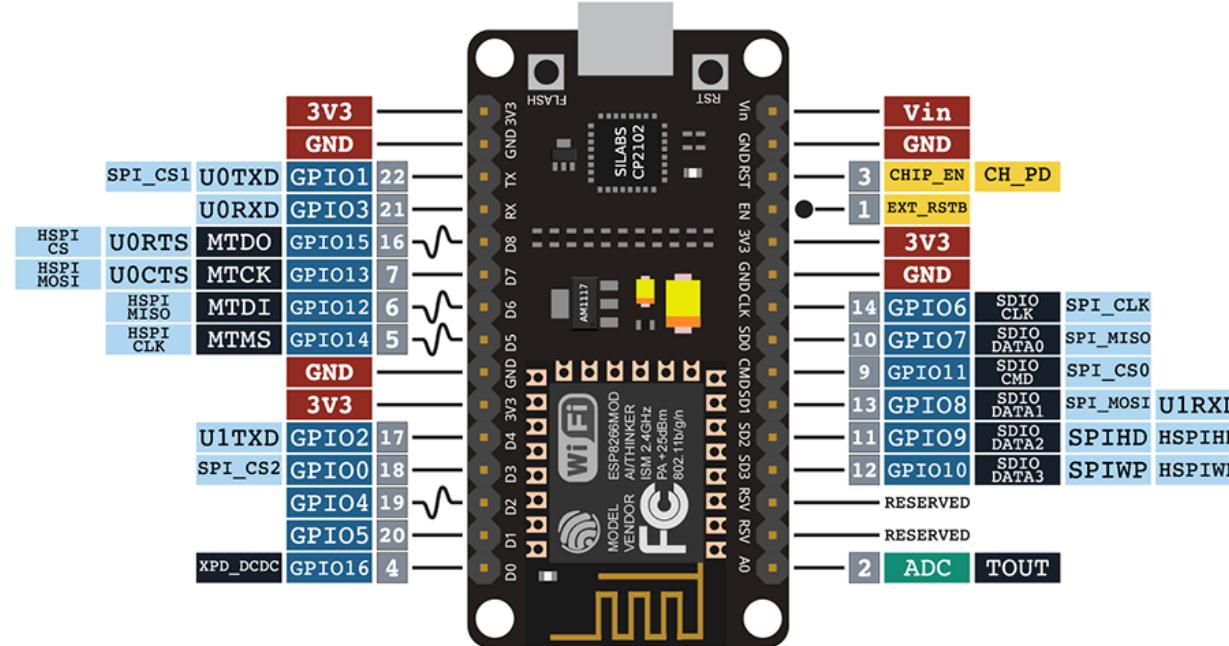
PINOUT

NOTES:

- ⚠ Typ. pin current 6mA (Max. 12mA)
- ⚠ For sleep mode, connect GPIO16 and EXT_RSTB. On wakeup, GPIO16 will output LOW for system reset.
- ⚠ On boot/reset/wakeup, keep GPIO15 LOW and GPIO2 HIGH.



FIWARE
ZONE



POWER	SP. FUNCTION(S)
I/O	COMM. INTERFACE
ADC	PIN NUMBER
CONTROL	PWM
N/C	

Telefonica



02 >

Instalación e IDE

2.1 Drivers



MacBook Pro

Árbol de dispositivos USB

- Receptor de infrarrojos
- Teclado/Trackpad interno de Apple
- Bus USB
 - CP2102 USB to UART Bridge Controller
 - Bus USB 2.0
 - Lector de tarjetas de memoria interna
 - Bus USB 2.0
 - iSight integrada

CP2102 USB to UART Bridge Controller:

ID del producto:	0xea60
ID del fabricante:	0x10c4 (Silicon Laboratories, Inc.)
Versión:	1.00
Número de serie:	0001
Velocidad:	Hasta 12 Mb/s
Fabricante:	Auto Formato
ID de la ubicación:	Archivo de programa.
Corriente dispuesta:	Reparar codificación & Recargar.
Corriente necesaria:	Monitor Serie
Corriente operativa:	Serial Plotter
WiFi101 Firmware Updater	
Placa:	"NodeMCU 1.0 (ESP-12E Module)"
Flash Size:	"4M (1M SPIFFS)"
Debug port:	"Disabled"
Debug Level:	"Ninguno"
lWIP Variant:	"v2 Prebuilt (MSS=536)"
CPU Frequency:	"80 MHz"
Upload Speed:	"115200"
Puerto:	"/dev/cu.SLAB_USBtoUART"
Obtén información de la placa	
Programador:	"AVRISP mkII"
Quemar Bootloader	

MBP > Hardware > USB > Bus USB > CP2102 USB to UART Bridge Controller

Device Manager

File Action View Help

- Bluetooth
- Computer
- Disk drives
- Display adapters
- DVD/CD-ROM drives
- Human Interface Devices
- IDE ATA/ATAPI controllers
- Imaging devices
- Jungo Connectivity
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- Other devices
- Ports (COM & LPT)
 - Silicon Labs CP210x USB to UART Bridge (COM3)
- Print queues
- Processors
- SD host adapters
- Software devices

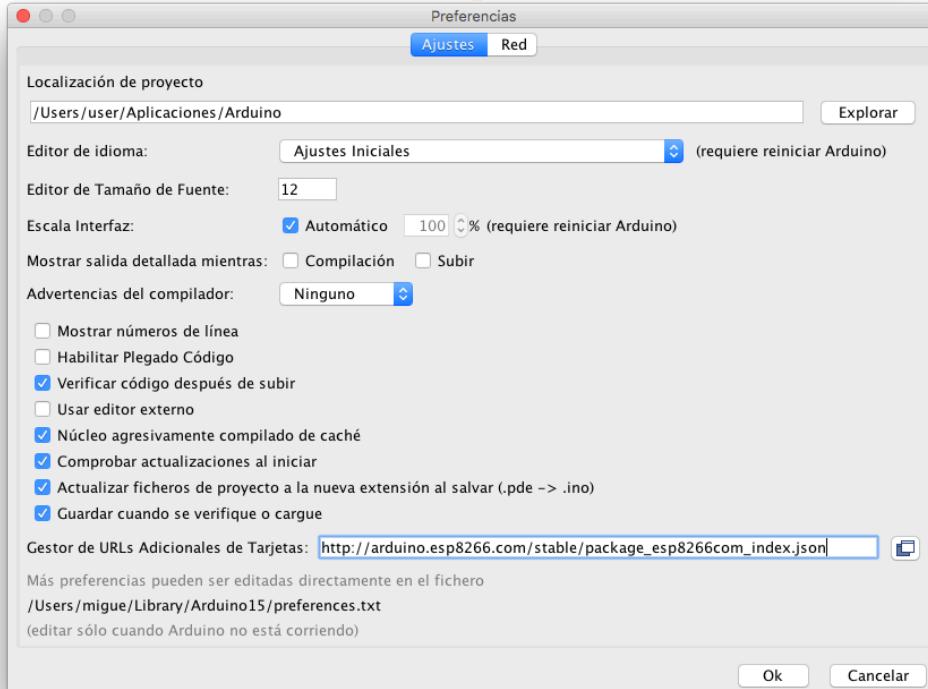
www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers

Telefónica

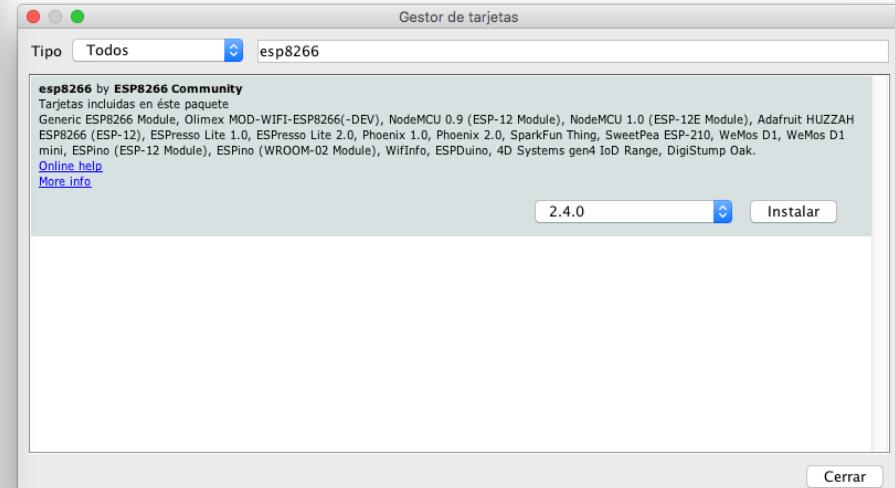
2.2 Arduino



1



2



http://arduino.esp8266.com/stable/package_esp8266com_index.json

2.3 Postman

A screenshot of the Postman application interface. On the left, there's a sidebar with a tree view of requests organized into folders. The main area shows a request configuration screen with a "Send" button. Below it is a code editor containing a script with environment variables and a test block. At the bottom, there's a results panel displaying a JSON response. Orange callout boxes with arrows point from the text to specific parts of the interface: one points to the sidebar with the text "Organize requests into folders.", another points to the code/test area with "Document the collection with descriptions, tests, and more.", and a third points to the results panel with "Examine responses and view results.".

Organize requests into folders.

Document the collection with descriptions, tests, and more.

Send requests individually, or use collection runner to send all the requests in the collection.

Examine responses and view results.

<https://www.getpostman.com>

2.4 Repositorio del curso



This repository Search Pull requests Issues Marketplace Explore

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

FIWARE IoT integration using ESP866 board Edit

Add topics

6 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

FIWAREZone Uploaded arduino libraries Latest commit 91b02ed a minute ago

Arduino Source Uploaded arduino libraries a minute ago

.DS_Store Uploaded arduino libraries a minute ago

README.md Update README.md 17 hours ago

README.md

IoT_Course

FIWARE IoT integration using ESP866 board

This course help developers to integrata a IoT development board by using NodeMCU with Arduino firmware. The communication protocol used by device is Ultralight2.0.

https://github.com/FIWAREZone/IoT_Course

03 >

Configurando
dispositivo en FIWARE

3.5 Creando la entidad

HTTP POST:

<https://{{host}}:10027/v2/entities>

Headers: {'content-type': 'application/json'; "Fiware-Service": {{service}}"; "Fiware-ServicePath":{{subservice}}"; "X-Auth-Token: {{token}}"}
Payload:

```
{  
    "id": "demoRoom",  
    "type": "Room",  
    "locked": {  
        "value": true,  
        "type": "Boolean"  
    },  
    "temperature": {  
        "value": 42.3,  
        "type": "float"  
    },  
    "humidity": {  
        "value": 82.1,  
        "type": "float"  
    }  
}
```

3.5 Creando la entidad

A screenshot of a web-based IoT platform interface. The title bar says "IOT Platform". The main area shows "Entidad: demoRoom". On the left is a sidebar with icons for entities, sensors, actuators, locations, and services. The main content has tabs "Gestión de Entidades" and "Detalle de Entidad".

DATOS BÁSICOS

ID	Tipo
demoRoom	Room

ATRIBUTOS

NOMBRE	TIPO	VALOR	METADATO
humidity	float	82.1	
locked	Boolean	true	
temperature	float	42.3	

Borrar △Precaución! Esta acción no se puede deshacer

3.5 Creando el dispositivo



HTTP POST:

https://{{host_iota}}:8088/iot/devices

Headers: {'content-type': 'application/json'; "Fiware-Service": **{{service}}**"; "Fiware-ServicePath":**{{subservice}}**"; "X-Auth-Token: **{{token}}**”}

Payload:

```
{  
  "devices": [  
    {  
      "device_id": "id_sensor_demo",  
      "entity_name": "demoRoom",  
      "entity_type": "Room",  
      "attributes": [  
        { "object_id": "t", "name": "temperature", "type": "float" },  
        { "object_id": "h", "name": "humidity", "type": "float" }  
      ],  
      "lazy": [ ],  
      "commands": [  
        { "object_id": "conf", "name": "conf", "type": "command" }  
      ],  
      "static_attributes": [ ],  
      "protocol": "IoTA-UL",  
      "transport": "HTTP"  
    }  
  ]  
}
```

3.5 Creando el dispositivo



The screenshot shows the IoT Platform interface for creating a new device. The URL in the browser is 195.235.93.224. The navigation path is: Gestión de dispositivos > IoTAgent Ultralight 2.0 - Node.js > Detalle de un dispositivo. The device is named "id_sensor_demo".

DATOS BÁSICOS

ID	Nombre De La Entidad	Tipo De Entidad
id_sensor_demo	demoRoom	Room

Entidad Relacionada
demoRoom/Room

MAPEO DE ATRIBUTOS

Valores por defecto

PARÁMETRO DE PROTOCOLO	ATRIBUTO DE DISPOSITIVO	TIPO

Valores establecidos para este dispositivo

PARÁMETRO DE PROTOCOLO	ATRIBUTO DE DISPOSITIVO	TIPO
t	temperature	float
h	humidity	float

3.5 Creando el dispositivo



The screenshot shows the IOT Platform interface on a Mac OS X desktop. The window title is "IOT Platform" and the sub-header is "Gestión de Entidades > Detalle de Entidad". The main content area displays an entity named "demoRoom" with the following details:

ID	Tipo
demoRoom	Room

Below this, under the "ATRIBUTOS" section, there is a table listing various attributes and their values:

NOMBRE	TIPO	VALOR	METADATO
TimeInstant	ISO8601		
conf_info	commandResult		
conf_status	commandStatus	UNKNOWN	
humidity	float		
locked	Boolean	true	
temperature	float		
conf	command		

A green button at the bottom right of the table says "enviar comando" (Send command).

3.5 Enviando datos por UL



HTTP POST:

`http://{{host_iota}}:8085/iot/d?k={{UL_apikey}}&i={{device_ID}}`

Headers: {'content-type': 'text/plain'}

Payload:

t|99#h|98

3.5 Enviando datos por UL



IOT Platform

Gestión de Entidades Detalle de Entidad

Entidad: demoRoom

refrescar editar

DATOS BÁSICOS

ID	Tipo
demoRoom	Room

ATRIBUTOS

NOMBRE	TIPO	VALOR	METADATO
TimeInstant	ISO8601	2018-04-05T11:26:03.00Z	
conf_info	commandResult		
conf_status	commandStatus	UNKNOWN	
humidity	float	98	TimeInstant: 2018-04-05T11:26:03.489Z
locked	Boolean	true	
status		FAIL	
temperature	float	99	TimeInstant: 2018-04-05T11:26:03.532Z
conf	command		

enviar comando

3.5 Enviando datos al dispositivo



HTTP POST:

https://{{host}}:10027/v1/contextEntities/type/{{device_type}}/id/{{entity_name}}/attributes/conf

Headers: {'content-type': 'application/json'; "Fiware-Service: {{service}}"; "Fiware-ServicePath:{{subservice}}"; "X-Auth-Token: {{token}}"}

Payload:

```
{"value":{"paramName":"paramValue"},"type":"command"}
```

3.5 Enviando datos al dispositivo



IOT Platform

Gestión de Entidades Detalle de Entidad

Entidad: demoRoom

refrescar editar

DATOS BÁSICOS

ID	Tipo
demoRoom	Room

ATRIBUTOS

NOMBRE	TIPO	VALOR	METADATO
TimeInstant	ISO8601	2018-04-05T11:27:39.00Z	
conf_info	commandResult		
conf_status	commandStatus	PENDING	TimeInstant: 2018-04-05T11:27:39.690Z
humidity	float	98	TimeInstant: 2018-04-05T11:26:03.489Z
locked	Boolean	true	
status		FAIL	
temperature	float	99	TimeInstant: 2018-04-05T11:26:03.532Z
conf	command		

enviar comando

3.5 Enviando datos al dispositivo



The screenshot shows the Postman application interface. The top navigation bar includes "New", "Import", "Runner", "My Workspace", "IN SYNC", and various status indicators. The main workspace shows a POST request to the URL `http://{{host_iota}}:8085/iot/d?k={{UL_apikey}}&i={{device_ID}}&getCmd=1`. The "Body" tab is selected, showing the raw JSON payload: `t|88#h|55`. Below the request, the response details are shown: Status: 200 OK, Time: 208 ms, Size: 520 B. The "Pretty" view of the response body highlights the parameter `i 1 id_sensor_demo@conf!paramName=paramValue`, which is circled in red.

3.5 Enviando datos al dispositivo



IOT Platform

Gestión de Entidades Detalle de Entidad

Entidad: demoRoom

refrescar editar

DATOS BÁSICOS

ID	Tipo
demoRoom	Room

ATRIBUTOS

NOMBRE	TIPO	VALOR	METADATO
TimeInstant	ISO8601	2018-04-05T11:28:05.00Z	
conf_info	commandResult		TimeInstant: 2018-04-05T11:28:05.178Z
conf_status	commandStatus	DELIVERED	TimeInstant: 2018-04-05T11:28:05.178Z
humidity	float	55	TimeInstant: 2018-04-05T11:28:05.122Z
locked	Boolean	true	
status		FAIL	
temperature	float	88	TimeInstant: 2018-04-05T11:28:05.055Z
conf	command		

enviar comando

04 >

Programa y
funcionamiento

4.2 Variables de configuración

```
***** Global Variables *****/
//FIWARE Variables
String FIWARE_ID = "REEMPLAZAR_ID";
String FIWARE_server = "195.235.93.235";
String FIWARE_port = "8085";
String FIWARE_token = "REEMPLAZAR_APIKEY";

//WiFi Network Names
const char* WiFi_SoftAP_Name = "FIWAREZone_IoT";
const char* WiFi_SoftAP_WiFi_Name = "FIWAREZone_IoT_Wifi";

//OTA
const char* update_path = "/webota";
const char* update_username = "admin";
const char* update_password = "admin";
```

4.2 Librerías empleadas

```
//ESP8266 Native libraries
#include <ESP8266WiFi.h>          //https://github.com/esp8266/Arduino
#include <DNSServer.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>
#include <ESP8266HTTPUpdateServer.h>
#include <ESP8266HTTPClient.h>

//External libraries
#include <WiFiManager.h>          //https://github.com/tzapu/WiFiManager
#include <ArduinoJson.h>            //https://github.com/bblanchon/ArduinoJson
```

CONTENTS:

[Installing](#)[Reference](#)[Libraries](#)[File system](#)[ESP8266WiFi](#)[OTA Updates](#)[PROGMEM](#)[Boards](#)[FAQ](#)[Exception causes](#)[Debugging](#)[Stack Dump](#)[Using with Eclipse](#)

- [!\[\]\(c04d8496ab32fe2ff7d70966bf4e751c_img.jpg\) ESP8266AVRISP](#)
- [!\[\]\(419e6f3aaa80f8697d862e41f1e2fa6d_img.jpg\) ESP8266HTTPClient](#)
- [!\[\]\(1e70fba34d6f0801f6d032d9b23a77f4_img.jpg\) ESP8266HTTPUpdateServer](#)
- [!\[\]\(c72589a7ca7f431e2e64b2e5b3435201_img.jpg\) ESP8266LLMNR](#)
- [!\[\]\(7f484132f2d769bb2be0fd9a5dabae9e_img.jpg\) ESP8266NetBIOS](#)
- [!\[\]\(c8bc642d50d402c4c160d3f518b7670d_img.jpg\) ESP8266SSDP](#)
- [!\[\]\(a4f44e76b66502e708e59789da4a0b4e_img.jpg\) ESP8266WebServer](#)
- [!\[\]\(d2ae181df1c61518d4c3ca08001b5d49_img.jpg\) ESP8266WiFi](#)
- [!\[\]\(e27e9f0ae30cb1bbbff30eb9b6254e28_img.jpg\) ESP8266WiFiMesh](#)
- [!\[\]\(542159ff48500bef30a79b509074e22a_img.jpg\) ESP8266httpUpdate](#)
- [!\[\]\(bc8cf86a9bd0ae99a299fa25941ba470_img.jpg\) ESP8266mDNS](#)

<https://github.com/esp8266/Arduino>

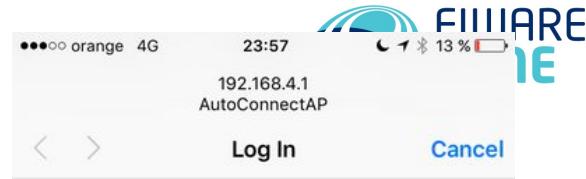
4.2.2 Librerías externas

WiFiManager

Permite configurar un punto de acceso
Si no lo encuentra, crea red propia para configurar
conectándose mediante wifi
Guarda las credenciales en la memoria interna

<https://github.com/tzapu/WiFiManager>

Telefónica



AutoConnectAP

WiFiManager

Configure WiFi

Configure WiFi (No Scan)

4.2.2 Librerías externas



ArduinoJson

Serialización y deserialización de JSON de forma sencilla para el envío de datos



<https://github.com/bblanchon/ArduinoJson>

Telefónica

4.3 Configuración WiFi



192.168.4.1

4.3 Configuración WiFi

Portal cautivo, entrar en “Configure WiFi” e introducir los datos de la red wifi



FIWARE-WiFi

WiFiManager

Configure WiFi

Configure WiFi (No Scan)

Info

Exit

192.168.4.1

Telefónica

4.3 Configuración WiFi



Screenshot of a web interface showing a list of available WiFi networks and their signal strengths. The IP address in the address bar is 192.168.1.168.

SSID	Signal Strength (%)
UVI24	62%
MOVISTAR_195E	54%
DIRECT-D0-HP OfficeJet Pro 8710	28%
MOVISTAR_F6A0	24%
Promalaga-UrbanLab-R1	24%
DIRECT-YMLAPTOP-HICCE8QDmsIE	22%
POLO-usuarios	22%
Promalaga-UrbanLab-R2	20%
Promalaga-UrbanLab_ Invitados	20%
Promalaga-UrbanLab-AV	18%
Moto G Play 5011	16%
Promalaga-UrbanLab_ Usuarios	14%

Below the list are two input fields: "SSID" and "password", each with a small "..." button to the right. At the bottom is a large blue "save" button.

[Scan](#)

4.3 Configuración WiFi



Revisamos el monitor del puerto serie
Ya está conectado a la Wifi que hemos
configurado. Ahora podemos acceder
desde el navegador, conectado a la misma
red

A screenshot of a terminal window titled "/dev/cu.SLAB_USBtoUART". The window shows a log of WiFi connection attempts and successful connections. A red box highlights the successful connection message. The terminal includes standard controls like 'Enviar' (Send), 'Autoscroll', and baud rate selection.

```
?  
=$??  
Dev-name: SC-1086424  
*WM:  
*WM: AutoConnect  
*WM: Connecting as wifi client...  
*WM: Using last saved values, should be faster  
*WM: Connection result:  
*WM: 3  
*WM: IP Address:  
*WM: 192.168.1.168  
Connected!  
mDNS responder started  
TCP server started
```

Enviar

Autoscroll Retorno de carro 9600 baudio Clear output

4.4 Código fuente



FIWARE.ino

```
void setup()
```

```
//Bind Webserver URL functions
server.on ( "/", handleRoot );
server.on ( "/wrst", handleWrst );
server.on ( "/wifi", handleWifi );
server.on ( "/webota", handleWebota );
server.on ( "/help", handleHelp );
server.on ( "/postul2", handlePostUL2 );
server.on ( "/postul2data", handlePostUL2data );
```

router.ino

Todas las funciones que renderizan las páginas

```
void handleRoot() {

    String page = FPSTR(HTTP_HEAD);
    page.replace("{v}", "FIWARE_Zone_IoT");
    page += FPSTR(HTTP_STYLE);
    page += FPSTR(HTTP_HEAD_END);
    page += F("<h1>FIWARE IoT Device</h1>");
    page += F("<h3>Menu</h3>");
    page += FPSTR(HTTP_MAIN_FORM);
    page += String("hostname:");
    page += String(dev_hostname);
    page += String(".local");
    page += FPSTR(HTTP_END);
    server.send(200, "text/html", page);

}
```

4.4.1 Página de inicio

router.ino

```
const char HTTP_MAIN_FORM[ ]
```

```
const char HTTP_MAIN_FORM[] PROGMEM = "<form action=\"/wifi\" method=\"get\"><button>Setup Wifi</button></form><br/><form action=\"/wrst\" method=\"get\"><button>Clear wifi setup</button></form><br/><form action=\"/webota\" method=\"get\"><button>Update</button></form><br/><form action=\"/postul2\" method=\"get\"><button>POST UL2</button></form><br/><form action=\"/help\" method=\"get\"><button>Help</button></form><br/>"
```

```
const char HTTP_MAIN_FORM[] PROGMEM =  
<form action="/wifi" method="get"><button>Setup Wifi</button></form><br/>  
<form action="/wrst" method="get"><button>Clear wifi setup</button></form><br/>  
<form action="/webota" method="get"><button>Update</button></form><br/>  
<form action="/postul2" method="get"><button>POST UL2</button></form><br/>  
<form action="/help" method="get"><button>Help</button></form><br/>
```

Utilidades de interés:

Convertidor string C (caracteres de escape)

http://tomeko.net/online_tools/cpp_text_escape.php?lang=en

Formatador de HTML

<http://minifycode.com/html-beautifier/>

4.4.1 Página de inicio



The screenshot shows a web browser window with the address bar displaying "192.168.1.168". The main content area is titled "FIWARE IoT Device" and contains a "Menu" section with five blue buttons:

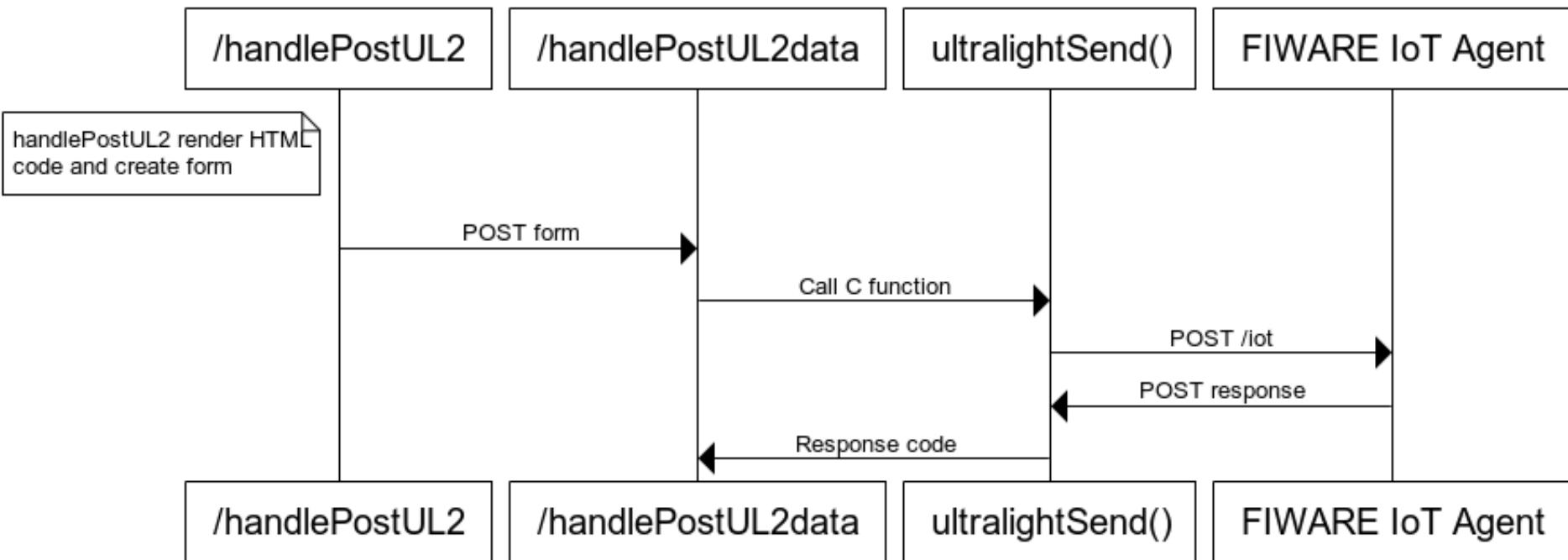
- Setup Wifi
- Clear wifi setup
- Update
- POST UL2
- Help

Below the menu, the text "hostname:SC-1086424.local" is displayed.

4.4.2 Envío por UL a FIWARE



Data sequence



www.websequencediagrams.com

4.4.2 Envío por UL a FIWARE

```
void handlePostUL2(){
    String page = FPSTR(HTTP_HEAD);
    page.replace("{v}", "Info");
    page += FPSTR(HTTP_SCRIPT);
    page += FPSTR(HTTP_STYLE);
    page += FPSTR(HTTP_HEAD_END);
    page += F("<h1>Enviar datos a FIWARE</h1>");
    page += F("<form action=\"/postul2data\" method=\"post\"> Temperature: <input type=\"number\"");
    page += FPSTR(HTTP_BACK_BUTTON);
    page += FPSTR(HTTP_END);
    server.send(200, "text/html", page);
}
```



```
<form action="/postul2data" method="post">Temperature:
<input type="number" name="tmp" min="-273" max="999">
<br>Relative Humidity:
<input type="number" name="rh" min="0" max="100">
<br>
<button type="submit" value="Submit">Submit</button>
</form>
```

4.4.2 Envío por UL a FIWARE

A screenshot of a web browser window titled "Enviar datos a FIWARE". The address bar shows the IP address "192.168.1.168". The page contains two input fields: "Temperature:" and "Relative Humidity:", each with a dropdown arrow icon. Below the inputs is a large blue "Submit" button. At the bottom of the page is a blue "Back" button.

4.4.2 Envío por UL a FIWARE

```
void handlePostUL2data(){  
    //Search form post values  
    String tmp, rh;  
    if (server.args() > 0 ) {  
        for ( uint8_t i = 0; i < server.args(); i++ ) {  
            if (server.argName(i) == "tmp") {  
                // do something here with value from server.arg(i);  
                tmp = server.arg(i);  
            }  
            else if(server.argName(i) == "rh"){  
                rh = server.arg(i);  
            }  
        }  
    }  
  
    //POST Data  
    String page = FPSTR(HTTP_HEAD);  
    page.replace("{v}", "Info");  
    page += FPSTR(HTTP_SCRIPT);  
    page += FPSTR(HTTP_STYLE);  
    page += FPSTR(HTTP_HEAD_END);  
    page += F("<h1>Enviendo datos a FIWARE</h1>");  
  
    //Parse response  
    int returnCode = ultralightSend(FIWARE_server,FIWARE_port,FIWARE_token,FIWARE_ID,"temp|"+tmp+"#hr|"+rh);  
    if (returnCode ==200){  
        page += F("<h3>Resultado</h3>");  
        page += F("La peticion se ha ejecutado correctamente");  
    }  
    else{  
        page += F("<h3>Resultado</h3>");  
        page += F("Ha habido un error en la peticion");  
        page += String(returnCode);  
    }  
  
    //End page  
    page += FPSTR(HTTP_BACK_BUTTON);  
    page += FPSTR(HTTP_END);  
    server.send(200, "text/html", page);  
}
```

4.4.2 Envío por UL a FIWARE

```
int ultralightSend (String URL, String port, String Token, String ID, String Body) {  
    int httpCode = 0;  
  
    Serial.println("http://" + URL + ":" + port + "/iot/d?k=" + Token + "&i=" + ID + "&getCmd=1");  
    http.begin("http://" + URL + ":" + port + "/iot/d?k=" + Token + "&i=" + ID + "&getCmd=1"); //Specify request destination  
  
    http.addHeader("Content-Type", "text/plain"); //Specify content-type header  
  
    httpCode = http.POST(Body); //Send the Body  
    String payload = http.getString(); //Get the response payload  
  
    Serial.println("Request Done");  
    Serial.print("Return code: "); //Print HTTP return code  
    Serial.println(httpCode);  
    Serial.print("Response payload: "); //Print request response payload  
    Serial.println(payload);  
  
    http.end(); //Close connection  
    return httpCode;  
}
```

4.4.2 Envío por UL a FIWARE



A screenshot of a web browser window titled "192.168.1.168". The main content area displays the heading "Enviando datos a FIWARE" in bold black font, followed by the section title "Resultado" in bold black font. Below this, the message "La peticion se ha ejecutado correctamente" is shown in black text. At the bottom of the content area is a blue rectangular button with the word "Back" in white.

4.5 OTA



```
//OTA
const char* update_path = "/webota";
const char* update_username = "admin";
const char* update_password = "admin";
```



4.5 OTA

